Programs from Art

Elodie Fourquet
Assistant Professor of Computer Science

Computers are ubiquitous in the everyday life of art and artists, from digitized art history and new media to the laptop on which I am writing this essay. They all depend on computer science, an applied discipline investigating the algorithms, data, networks, and systems from which practical tools are built—tools that help experts in other disciplines to solve problems. In this sense, computer science is necessarily interdisciplinary. Every computer scientist must be capable of comprehending the essence of the domain in which her products will be used. Thus, computer science responds vigorously to the broad expectations we have of a liberal arts education.

From introductory programming to specialized advanced courses, students need a creative framework that engages them and releases their open-ended personal ambitions. In the past I had students write programs that drew images based on works of art of their choice. Figure 1 is a composition by Thu Truong '14 (Mount Holyoke College), written in the Processing programming language, based on *Nighthawks* (1942) by Edward Hopper (American, 1882–1967)—her first-ever programming assignment.

When students start this process, they must reflect on their chosen artwork, think, and plan before programming. They begin with pencil and paper, acquiring ownership of the image, enhancing their commitment to the process, and strengthening their experience of art and its relationship to their scientific practice. Encouraging students to practice and appreciate contemplative work—a challenge for novice programmers, who are inclined to frantically rush coding as quickly as possible and without pause—makes them better programmers. Innovative practitioners of computer science must stand back from their work to think and rework their software designs—like Michelangelo, chisel in hand, standing back from the half-finished *David*, or Jackson Pollock taking a moment to refect before adding another layer of paint on his canvas.

Programming shares an interesting relationship with composition, art history, and studio technique. All of these areas are primarily learned by doing: in art history, "looking" is "doing." Incorporating art into the "doing" of computer science is natural and beneficial. In addition, both computer science and art encourage learning from others, taking inspiration from individual experience and sensibility. Since time immemorial artists have been inspired by nature, creating images and artifacts that embody the inner lives of individuals and the common life of society. The rhinoceros that first appeared in the Lascaux cave paintings has been the subject of a woodcut by Albrecht Dürer, paintings by Salvador Dalí, and a red resin sculpture by Xavier Veilhan (French, born 1963), shown today at the Pompidou Center in Paris.

FIG. 1 | Composition by Thu Truong '14 (Mount Holyoke College) based on *Nighthawks* (1942) by Edward Hopper

A liberal arts education aims to mediate between today's technology and the spirit of humanism. In my classroom, where I integrate art into computer science, students benefit from easy access to the Picker Art Gallery and its collection. Encountering original works in the Picker is a much richer and more satisfying experience than viewing art through books, reproductions, or the Internet. The gallery environment promotes the contemplative attitude that opens one's mind to a work of art. It also encourages an intimate exploration of the surface of a work of art, an element celebrated by many contemporary painters.

To take advantage of this rich resource in a first-year seminar I will teach next year, I have asked four computer science students to write programs responding to paintings from the Picker collection. Nianyi Wang '18 and Jingxian Wu '18 wrote programs in response to Alex Katz's *Late July 2* (1971) and *Moose* (1983), respectively, while Samantha Braver '18 and Lillian Pentecost '16 engaged with works by Alice Neel and Adolph Gottlieb.

.

CAT. 5

Alice Neel (American, 1900–1984)

Girl in Pajama Suit, 1945

Oil on canvas

22½ × 26½ in. (57.2 × 67.3 cm)

Gift of Edwin J. Safford '58, 1991.49'

Girl in Pajama Suit is an oil painting by Alice Neel, who is well-known as a portraitist. I was initially drawn to the work by its colors: red, mauve, and periwinkle blue. On second look, I was also intrigued by the expression on the girl's face.

In building a computer-generated image that responds to Neel's painting I faced two creative decisions. First, the meaning and power of most figurative painting emerges from a wealth of subtle details, a wealth that risks seeming ironic when simulated by a computer. Finding a level of abstraction appropriate to the available programming tools was a challenge. Does a single triangle suffice to represent an arm? Or must I build it from multiple primitives, or perhaps more complex rounded objects? How will the perception of the work change if I include eyes?

Such questions led inevitably to my second challenge: How should I combine the characteristics that are easily manipulated using a computer—color, shape, and level of detail—to interpret of the original painting? I was inspired to explore these pictorial aspects through my reading of Molly Bang's book *Picture This: How Pictures Work.* (New York: Chronicle, 2000).

Observing the original painting in the Picker Art Gallery collection added more dimensions to my experience of the picture—and to my ambition in responding to it. Being able to work closely with the painting, I was able to examine its true colors, brush strokes, and textures. This contemplation broadened my appreciation for the work and the project as a whole.

—SAMANTHA BRAVER '18

Copyright image removed

CAT. 6

Adolph Gottlieb (American, 1903–1974)

Orange Glow, 1970

Acrylic on paper

23¾ × 18¾ in. (60.3 × 47.6 cm)

Gift of Dr. Luther W. Brady H'88, 1992.59

Orange Glow by Adolph Gottlieb first presents a simplicity, but a simplicity that is illusionary. Further observation of the painting allows us to see the complexity of its formal elements.

The painting is a challenge for a student creating a computer-generated image inspired by it. The artwork appears simple and abstract in color and form but is actually composed of nongeometric textures and shapes with irregular boundaries. These features make the work very difficult to recreate with computer graphics techniques. For example, the irregular lines surrounding the upper circle in the painting are too regular to be generated by random numbers but too irregular to be generated by a precise, iterative method that would calculate a predictable distribution of shapes.

Gottlieb's lines and shapes are tangled and organic, intentionally unclear and irregular. Imposing straight lines or uniform texturing, which are easy to do with most computer graphics tools, would rob the work of its emotional power if done insensitively. Thus, intensive study of the work was necessary to understand its composition, particularly the artist's use of underpainting, before attempting a program in response. I made creative decisions to simplify and reinterpret *Orange Glow* so that the overall effect, texture, and pattern of the painting were retained. This example highlights how a programmer who is aware of artistic considerations and mindful of more than just the physical appearance of a work may produce a successful reproduction and nurture artistic appreciation otherwise not present in computer science curricula.

—LILLIAN PENTECOST '16

Copyright image removed